Opie, George L.

From: Roccia, Vincent J. (Woodcock Washburn) [vroccia@woodcock.com]

Sent: Thursday, February 09, 2006 10:26 AM

To: Opie, George L.

Subject: 09/240,406/ DOCKET NO.: MSFT-0510/36349.1

Examiner Opie,
Attached is a copy of the amended claims as you suggested for the Examiner's amendment. Give me a call if you have any questions.

Thanks
Vince Roccia
****

1-15. (Canceled)
16. (Currently Amended) A system for extending functionality of a class object, comprising:
a processing unit;
a system memory in communication with the processing unit via a system bus;
a computer-readable medium in communication with the processing unit via the system bus; and
an extensible object model executed from the computer-readable medium by the processing unit, wherein the extensible object model determines whether a requested functionality is inherent in the class object, and wherein the extensible object model causes the processing unit to create an extension object from an existing extension package when a requested functionality is not inherent in the class object, and wherein the extension object extends the class object to provide the requested functionality~~.~~, and wherein the extensible object model further causes the processing unit to notify the extension object when the extension is deleted, and wherein the extensible object model further causes the processing unit to register the extension package in an extension database stored on the computer-readable medium, and wherein the extensible object model further causes the processing unit to store the extension object in system memory when the corresponding extension is first referenced, and wherein the extensible object model further causes the processing unit to create the extension object from the extension package by causing the processing unit to create an extension provider object and causes the processing unit to create the extension from the extension provider object.

17-20 (Cancelled)
21. (Original) The computerized system of claim 16, wherein the extensible object model further causes the processing unit to create an event filtering and sourcing object to handle events generated by the extension object.

22-26. (Canceled)
27. (Currently Amended) A computer-readable medium having stored thereon computer-executable components comprising:
an extensible object;
an extension database having an entry for an extension for the extensible object; and
an existing extension package having an interface for obtaining an extension object that provides the extension for the extensible object, wherein the extensible package determines whether a requested functionality is inherent in the class object~~.~~;

an extensible object model, wherein the extensible object model further causes the processing unit to notify the extension object when the extension is deleted, and wherein the extensible object model further causes the processing unit to register the extension package in an extension database stored on the computer-readable medium, and wherein the extensible object model further causes the processing unit to store the extension object in system memory when the corresponding extension is first referenced, and wherein the extensible object model further causes the processing unit to create the extension object from the extension package by causing the processing unit to create an extension provider object and causes the processing unit to create the extension from the extension provider object.

28. (Original) The computer-readable medium of claim 27, further comprising:
an extension provider object that proffers the extension object as a result of a call to the interface in the extension package.

29. (Currently Amended) A method for extending functionality of a class object in a run-time environment, comprising:

determining whether a requested functionality is inherent in the class object;
receiving a request from an application for functionality that is not inherent in the class object;
determining if the functionality is available in a first extension object;
obtaining an existing extension package having computer-executable instructions associated with the extension object functionality, wherein the extension package proffers an extension provider object when the functionality is requested~~;~~, and wherein the extensible object model further causes the processing unit to notify the extension object when the extension is deleted, and wherein the extensible object model further causes the processing unit to register the extension package in an extension database stored on the computer-readable medium, and wherein the extensible object model further causes the processing unit to store the extension object in system memory when the corresponding extension is first referenced, and wherein the extensible object model further causes the processing unit to create the extension object from the extension package by causing the processing unit to create an extension provider object and causes the processing unit to create the extension from the extension provider object.

specifying parameters to the extension provider object to create a second extension object; and

directing the request to the second extension object.

30. (Previously presented) The method of claim 29, further comprising registering the extension package in an extension database.

31. (Previously presented) The method of claim 29, further comprising storing the extension package in an extension database.

32. (Previously presented) The method of claim 31, further comprising searching for an entry associated with the functionality in the extension database to determine if the functionality is available in the extension object.

33. (Previously presented) The method of claim 29, further comprising creating the second extension object when the extended functionality is first referenced, and locating the second extension object when the extended functionality is subsequently referenced.

34-44 (Cancelled)